IN THE CLAIMS

1. (original) A method for modifying a program to allow the program to execute on a processor system that includes a programmable logic device, the method comprising:

identifying a critical code segment of the program;

rewriting the critical code segment as a function;

revising the program by designating the function as a code to be compiled by an extension compiler and by replacing the critical code segment with a statement that calls the function; and

compiling the revised program such that the function is executed by the programmable logic device.

2. (original) The method of claim 1 wherein the critical code segment is defined by the length of time required for execution.

3. (original) The method of claim 1 wherein the critical code segment is a nested loop.

4. (original) The method of claim 1 wherein the program is written in a programming language and the function is written with the same programming language.

5. (original) The method of claim 1 wherein the function is selected from a library of pre-defined functions.

6. (original) The method of claim 1 wherein the function defines an integer with a non-standard number of bits.

7. (original) The method of claim 1 wherein the program is written in a program file and designating the function as a code includes writing the code to an extensions file.

8. (original) The method of claim 1 wherein compiling the revised program includes copying the code to an extensions file.

9. (original) The method of claim 1 wherein compiling the revised program includes compiling an extensions file including the code to produce a header file and an intermediate file written in a hardware description language.

10. (original) The method of claim 1 wherein the step of revising is performed manually.

11. (original) The method of claim 1 wherein the step of revising is performed using an automated conversion tool.

12. (original) The method of claim 9 wherein the hardware description language is Verilog HDL.

13. (original) The method of claim 9 wherein the header file declares a prototype for the function.

14. (original) The method of claim 9 wherein the intermediate file includes an implementation of the function as an instruction for a programmable logic device.

15. (original) The method of claim 10 wherein the header file and the revised program are compiled together by a standard compiler to generate an executable file.

16. (original) The method of claim 15 wherein the standard compiler also includes the compiling of a configuration file in generating the executable file.

17. (original) The method of claim 1 further comprising:

profiling the revised program; and

evaluating the performance of the revised program.

18. (original) The method of claim 17 wherein evaluating the performance of the revised program includes comparing the performance against a timing requirement.

19. (original) The method of claim 17 wherein evaluating the performance of the revised program includes comparing the performance against a prior performance.

20. (original) The method of claim 1 wherein the function executed by the programmable logic device does not have direct access to non-register file memory.

21. (original) The method of claim 1 wherein the function executed by the programmable logic device has register file inputs and outputs limited to a predetermined number set by the compiler.

22. (original) The method of claim 21 wherein the limited predetermined number of register file inputs is three.

23-37. (withdrawn)

38. (original) A method for extending the native instruction set of a general purpose processor in a computing system comprising a general purpose processor and a programmable logic device, the method consisting of the steps of:

(i) identifying critical code segments in an application program to be run on the computing system;

(ii) replacing the critical code segments with at least one extended instruction, not included in the native instruction set of the processor;

(iii) compiling the application program including the code segments containing the extended instruction; and

(iv) executing the compiled application program on the computer system such that the native instructions are executed by the processor and the extended instruction is executed by the programmable logic device.

39. (original) The method of claim 38 wherein the critical code segment is defined by the length of time required for execution.

40. (original) The method of claim 38 wherein the critical code segment is a nested loop.

41. (original) The method of claim 38 wherein the at least one extended instruction is selected from a library of predefined extended instructions.

42. (original) The method of claim 38 wherein compiling the application program includes copying the application program to an extensions file.

43. (original) The method of claim 38 wherein compiling the application program includes compiling an extensions file including the code to produce a header file and an intermediate file written in a hardware description language.

44. (original) The method of claim 43 wherein the hardware description language is Verilog HDL.

45. (original) The method of claim 38 wherein the step of revising is performed manually.

46. (original) The method of claim 38 wherein the step of revising is performed using an automated conversion tool.

47-53. (withdrawn)

54. (original) A system for modifying a program to allow the program to execute on a processor system that includes a programmable logic device, comprising:

means for identifying a critical code segment of the program;

means for rewriting the critical code segment as a function;

means for revising the program by designating the function as a code to be compiled by an extension compiler and by replacing the critical code segment with a statement that calls the function; and

means for compiling the revised program such that the function is executed by the programmable logic device.

55. (original) The system of claim 54 wherein the critical code segment is defined by the length of time required for execution.

56. (original) The system of claim 54 wherein the critical code segment is a nested loop.

57. (original) The system of claim 54 wherein the program is written in a programming language and the function is written with the same programming language.

58. (original) The system of claim 54 wherein the function is selected from a library of pre-defined functions.

59. (original) The system of claim 54 wherein the function defines an integer with a non-standard number of bits.

60. (original) The system of claim 54 wherein the program is written in a program file and means for revising the program by designating the function as a code includes means for writing the code to an extensions file.

61. (original) The system of claim 54 wherein the program is written in a program file and means for revising the program by designating the function as a code includes means for writing the code into the program file and demarking the code.

62. (original) The system of claim 54 wherein means for compiling the revised program includes means for copying the code to an extensions file.

63. (original) The system of claim 54 wherein means for compiling the revised program includes means for compiling an extensions file including the code to produce a header file and an intermediate file written in a hardware description language.

64. (original) The system of claim 63 wherein the hardware description language is Verilog HDL.

65. (original) The system of claim 63 wherein the header file declares a prototype for the function.

66. (original) The system of claim 63 wherein the intermediate file includes an implementation of the function as an instruction for a programmable logic device.

67. (original) The system of claim 63 wherein the header file and the revised program are compiled together by a standard compiler to generate an executable file.

68. (original) The system of claim 54 further comprising:

means for profiling the revised program; and

means for evaluating the performance of the revised program.

69. (original) The system of claim 68 wherein the means for evaluating the performance of the revised program includes means for comparing the performance against a timing requirement.

70. (original) The system of claim 68 wherein the means for evaluating the performance of the revised program includes means for comparing the performance against a prior performance.

71. (original) The system of claim 54 wherein the function executed by the programmable logic device does not have direct access to non-register file memory.

72. (original) The system of claim 54 wherein the function executed by the programmable logic device has register file inputs and outputs limited to a predetermined number set by the compiler.

73. (original) The system of claim 54 wherein the limited predetermined number of register file inputs is three.

74. (original) A method for modifying a program to allow the program to execute on a processor system that includes a programmable logic device, the method comprising:

identifying a critical code segment of the program;

demarking the critical code segment;

revising the program by designating the demarked code segment as a code to be compiled by an extension compiler and by replacing the critical code segment with one or more extended instructions; and

compiling the revised program such that the extended instructions are executed by the programmable logic device.